📒

포팅메뉴얼 - 정리 문서

프론트엔드

[사용자/그린메이트] 프로젝트 실행

npm

- v9.5.1

node

- v18.16.1

사용자 프로젝트 실행

```
> git clone https://lab.ssafy.com/s09-webmobile1-sub2/S09P12C202.git
(clone된 디렉토리 경로에서 VSC 실행)
> cd ./frontend
> npm install
> npm run start
```

그린메이트 프로젝트 실행

```
> git clone https://lab.ssafy.com/s09-webmobile1-sub2/S09P12C202.git
(clone된 디렉토리 경로에서 VSC 실행)
> cd greenmate/planty-greenmate
> npm install
> npm run start
```

[사용자/그린메이트] UI 라이브러리

UI 자바스크립트 라이브러리로써 싱글 페이지 애플리케이션(SPA)의 사용자 인터페이스(UI)를 생성하는 컴포넌트 기반의 라이브러리

React

- v 18.2.0

[사용자/그린메이트] 프로젝트 생성 및 빌드 환경 세팅

React 프로젝트를 생성하고, webpack, babel 등 프로젝트 빌드에 필요한 기본 설정을 수행.

create-react-app (CRA)

- v5.0.1
- 설정

  ```
  - project-name : planty
  - custom-templates : typescript
  - 기본 절대경로값 : tsconfig.json의 baseUrl
  ```

- 프로젝트 구조

  ```
  '/'
  | .env
  ```

```
| .eslintrc.json
| .prettierrc
| package-lock.json
| package.json
| README.md
| tsconfig.json
├─.vscode
| settings.json
├─node_modules
├─public
| └─icons
└─src
| App.tsx
├─assets
| ├─fonts
| ├─icons
| └─images
├─components
| ├─atoms
| ├─layout
| └─organisms
├─constants
├─hooks
| ├─api
| └─common
├─recoil
├─router
| AppRouter.tsx
| PrivateRoute.tsx
├─styles
| | index.scss
| ├─base
| └─mixins
├─types
| ├─common
| └─domain
└─utils
├─api
└─common
```

[사용자/그린메이트] 코드 정적 분석 및 포매팅

팀원 간 코드 스타일을 표준화하고, 더 가독성 있고 효율적인 문법으로 포매팅.

esLint

- v8.45.0
- 설정

```
- package.json의 'eslintConfig'을 제거.
- npx eslint --init 대화형 명령으로 .eslintrc.json 파일 추가
- airbnb 규칙 적용을 위한 .eslintrc.json 수정 (extends, parserOptions, plugins, rules 블록 수정)
```

- 관련 config 및 plugin 모듈

```
"eslint-config-airbnb": "19.0.4",
"eslint-config-airbnb-typescript": "17.1.0",
"eslint-config-prettier": "8.8.0",
"eslint-plugin-prettier": "5.0.0",
"eslint-plugin-react": "7.33.0",
"@typescript-eslint/eslint-plugin": "6.1.0",
"@typescript-eslint/parser": "6.1.0"
```

prettier

- v3.0.0
- 설정

```
- package.json 내 'scripts' 블록 수정
- .prettierrc.json 파일 추가
- .eslintrc.json 수정
- vscode - save on format 설정 또는 .vscode > settings.json 추가
```

[FE/GM] 컴파일러

- 타입스크립트 -> 자바스크립트 트랜스파일링 및 정적 분석을 위한 컴파일러

typescript

- v4.9.5

[사용자/그린메이트] @types 라이브러리

typescript 프로젝트에 맞게, 패키지 및 라이브러리의 타입을 추론하도록 돕는 보조 라이브러리.

- 관련 패키지

```
"@types/react": "^18.2.15",
"@types/react-dom": "^18.2.7",
"@types/react-lottie": "1.2.6",
"@types/react-slick": "^0.23.10",
"@types/swiper": "^6.0.0",
```

[사용자/그린메이트] 스타일링 (CSS)

`nesting`, `mixin` 등 스타일(css) 코드의 효율을 높이기 위한 전처리기/라이브러리

Sass

- v1.64.1
- CSS 전처리기
- 설정

```
- 공통 코드를 _common.scss에 작성.
- 자주 사용되는 mixin은 mixins 디렉토리에 별도로 관리
- 자주 사용되거나 변화할 가능성이 큰 속성 값은 _variable.scss에서 일괄적으로 관리
```

MUI

- v5.14.4
- React UI 라이브러리
  - 모달, 아코디언 등을 가져와서 공통 컴포넌트 구현에 활용
- 관련 라이브러리

```
"@emotion/react": "^11.11.1",
"@emotion/styled": "^11.11.0",
```

[사용자/그린메이트] 주요 라이브러리

axios

- v1.4.0
- ajax 비동기 통신 라이브러리
- 설정

```
- instance.ts 파일 추가 및 axios 인스턴스 정의 및 기본 설정
  (공통 header, JWT Token 통신 로직, axios interceptor 등)
- 도메인 별 .ts 파일 생성하여 모듈화
```

recoil

- v0.7.7
- 전역 상태관리 라이브러리
- 설정법

```
1. 사용자
도메인별 관리해야할 전역 상태가 많아서, 각각의 도메인 파일에서 관리.
도메인 별 .ts 파일 생성하고, 각 도메인에 관련된 전역 상태의 atom만 정의

2. 관리자
관리해야할 전역 상태가 많지 않아서, 하나의 파일에서 관리.
store.ts 파일 생성하고, 프로젝트에서 관리해야 하는 모든 전역 상태의 atom을 정의
```

react-router-dom

- v6.14.2
- 리액트 앱에서 사용되는 라우터 및 SPA 구현에 활용
- 설정값

```
- AppRouter.tsx 파일 추가 및 코드 작성 (BrowserRouter, Routes)
- App.tsx에서 AppRouter import
```

openvidu-browser

- v2.28.0
- `openVidu API`를 활용하기 위한 라이브러리

[사용자/그린메이트] 기타 라이브러리

```
"@react-oauth/google": "^0.11.1",
"chart.js": "4.3.3",
"react-chartjs-2": "5.2.0",
"react-calendar": "^4.6.0",
"jwt-decode": "3.1.2",
"moment": "^2.29.4",
"react-confirm-alert": "^3.0.6",
"react-hot-toast": "^2.4.1",
"react-lottie": "1.2.3",
"swiper": "^6.8.4",
"react-slick": "^0.29.0",
"slick-carousel": "^1.8.1",
"react-spinners": "^0.13.8",
"react-uuid": "2.0.0",
"classnames": "^2.3.2",
```

백엔드

[planty-service] 프로젝트 실행

IntelliJ IDE

- v2023.1.3

JDK

- Azul Zulu 17.42.19

Database 설정 (MySQL이 설치되었다고 가정)

```
Windows termial 실행
> `mysql -u root -p`
> `{MySQL root password}`
MySQL CLI 접속
mysql > create user planty@'%' identified by 'planty202'
mysql > create database planty default character set utf8;
mysql > GRANT ALL PRIVILEGES ON planty.* TO planty@'%' IDENTIFIED BY 'planty202';
mysql > source {local path}/S09P12C202/backend/planty-service/src/main/resources/assets/db/planty_ddl.sql
mysql > source {local path}/S09P12C202/backend/planty-service/src/main/resources/assets/db/planty_dml.sql
```

Spring Server 실행

```
IntelliJ 에서 `File` - `Open` - `S09P12C202/backend/planty-service/` 선택
`Run Configurations` - `PlantyServiceApplication` 선택
`/S09P12C202/backend/planty-service/src/main/resources` 경로에 `application-env.yml` 붙여넣기
`Run` 실행
```

DB 덤프 파일

- **planty_ddl.sql**

  planty_ddl.sql 다운로드

  > planty_ddl.sql

- **planty_dml.sql**

  planty_dml.sql 다운로드

  > planty_dml.sql

[BE] 빌드 환경

Build Tools

- Gradle v8.2.1

FrameWork

- SpringBoot v2.7.14

ORM

- Spring Data JPA

ETC

- Spring Security
- JWT
- Swagger 2.9.2

프로젝트 구조

```
└── main
    ├── generated
    ├── java
    │   └── com
    │       └── planty
    │           ├── api  /* REST API 요청관련 컨트롤러, 서비스, 요청/응답 모델 정의*/
    │           │   ├── booking
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── consulting
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── embedded
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── emergency
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── gm
    │           │   │   ├── account
    │           │   │   │   ├── controller
    │           │   │   │   ├── response
    │           │   │   │   └── service
    │           │   │   ├── booking
    │           │   │   │   ├── controller
    │           │   │   │   ├── response
    │           │   │   │   └── service
    │           │   │   ├── consulting
    │           │   │   │   ├── controller
    │           │   │   │   ├── request
    │           │   │   │   └── service
    │           │   │   ├── emergency
    │           │   │   │   ├── controller
    │           │   │   │   ├── request
    │           │   │   │   └── service
    │           │   │   ├── subscribe
    │           │   │   │   ├── controller
    │           │   │   │   ├── response
    │           │   │   │   └── service
    │           │   ├── payment
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── service
    │           │   ├── subscribe
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── subscribeProduct
    │           │   │   ├── controller
    │           │   │   └── request
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── ticketProduct
    │           │   │   ├── controller
    │           │   │   └── response
    │           │   │   └── service
    │           │   ├── user
    │           │   │   ├── controller
    │           │   │   └── model
    │           │   │   │   ├── request
    │           │   │   │   ├── response
    │           │   │   │   └── service
    │           ├── common /* 공용 유틸, 응답 모델, 인증, 예외처리 관련 정의*/
    │           │   ├── enums
    │           │   ├── exception
    │           │   │   └── handler
    │           │   ├── handler
    │           │   ├── interceptor
    │           │   ├── jwt
    │           │   ├── model
    │           │   ├── uitl
    │           ├── config /* WebMvc 및 JPA, Security, Swagger 등의 추가 플러그인 설정 정의*/
    │           └── db /* 디비에 저장될 모델 정의 및 쿼리 구현 */
    │           │   ├── entity
    │           │   │   ├── common
    │           │   └── repository
```

```
    │           ├── PlantyServiceApplication.java
    └── resources
        ├── assets
        │   └── db
        │       ├── insert_dummy_data.sql
        │       └── planty_ddl.sql
        ├── application.yml /* 웹 리소스(서버 host/port, DB) 관련 설정 정의 */
        └── application-env.yml /* 보안 웹 리소스(Spring security, DB, OpenVidu 계정 정보) 관련 설정 정의 */
```

Middleware

디바이스 정보

Arduino

- Arduino IDE 2.1.1
  - ArduinoJson 6.21.0
  - DHT sensor library 1.4.4
- NodeMCU 1.0 (ESP-12E Module)